## REMARKS

Claims 1 – 13 are now pending in the application. Applicant respectfully requests the Examiner to reconsider and withdraw the rejection(s) in view of the amendments and remarks contained herein.

### REJECTION UNDER 35 U.S.C. § 102

The Examiner rejected claims 1 – 13 under 35 U.S.C. § 102(e) as being anticipated by Wallach et al. (U.S. 6,179,486). Applicant respectfully traverses this rejection.

The present invention is directed to computers having operating systems that utilize firmware to access base address registers ("BARs"). The term "firmware" is a term of art in computers. Techdictionary.com defines firmware as:

> "Software stored in ROM or PROM; essential programs that remain even when the system is turned off. Firmware is easier to change than hardware but more permanent than software stored on disk."

As noted in the present application, all operating systems do not use firmware to write to BARs, with some operating systems directly accessing the BARs as opposed to using firmware to do so. [Application, par. 0007]

When a PCI slot fails, operating systems that utilize firmware to access BARs have had problems in determining which PCI slot failed because the PCI resource allocation map that is generated during boot-up may have been changed by the operating system, such as by the operating system changing the BAR values or updating the PCI resource allocation map in the event of a PCI configuration space transaction or hot plug operation. While the firmware may have the initial information contained in the PCI resource allocation map, this information may be outdated as discussed and the firmware thus no longer has accurate information from which to determine the failed PCI slot.

The present invention solves this problem. It does so by maintaining a PCI resource allocation map with firmware. Each time the operating system executes firmware to perform PCI configuration space transactions or PCI hot plug operations, this firmware maintained

PCI resource allocation map is updated. When a PCI error occurs and a PCI host bridge logs the failing address, the firmware then uses this firmware maintained PCI resource allocation map to determine which PCI slot corresponds to the failure.

Claim 1 is directed to a method of identifying a failing PCI slot in a computer that accesses base address registers with firmware. It requires creating a firmware maintained PCI resource allocation map, updating the firmware maintained PCI resource allocation map upon the occurrence of the firmware being called to execute at least one of a hot plug operation and a PCI configuration space transaction, and upon the host bridge logging an error address due to a failing PCI slot, identifying the failing PCI slot from the information in the firmware maintained PCI resource allocation map.

The Examiner takes the position that Wallach et al meets all the limitations of claim 1, apparently taking the position that because Wallach et al.'s configuration manager 500 uses software for maintaining a PCI resource allocation map, this *ipso facto* means that it does so with firmware. According to the Examiner:

> Contrary to Applicant's argument, Wallach discloses "the system components of the NetWare Operating System and an embodiment of the software components of the invention. A configuration manager 500 is responsible for managing all or some of the adapters on the PC buses 234 and 236 (FIG. 2) or 250, 252, 254 and 256 (FIG. 3). The configuration manager 500 keeps track of the configuration information for every managed adapter located on the fault tolerant computer system 100. The configuration manager 500 also allocates resources for every managed adapter and initializes each managed adapter's registers during a hot swap operation." See at least column 9, lines 10 – 23. . . . The configuration manager 500 uses the variable "board_num" when requesting information from the NetWare Operating System driver configuration tables." Thus, it is clear that in Wallach, the PCI resource allocation map is maintained by software, and software is also used for accessing the BARs. [Official Action mailed November 24, 2004, pp. 6 - 7]

Applicant submits that just because Wallach et al. discloses using software to maintain a PCI resource allocation map and also using software to access the BARs does not mean that it discloses using firmware to maintain a PCI resource allocation map and using firmware to access BARs. As discussed above, firmware is a particular type of software --

software that is stored in a read-only-memory or a programmable read-only-memory. Wallach et al. not only does not mention firmware, the references in Wallach et al. to the use of the NetWare operating system, as cited by the Examiner, likely show that Wallach et al. is the type of computer system where the operating system accesses the BARs directly and not one where firmware is used to access the BARs.

Further, it is axiomatic that the disclosure of a genus does not anticipate each species in the genus. Firmware is a species of the genus software. Thus, the disclosure in Wallach et al. that software is used to maintain a PCI resource allocation map and that software is used to access the BARs does not anticipate claim 1 which requires that a particular type of software -- firmware -- be used.

The present invention solves a problem in computer systems having operating systems that use firmware to access base address registers and Wallach et al. does not even discuss the use of firmware to access base address registers, let alone the aforementioned problem or the solution to this problem. Applicant submits that Wallach et al. fails to anticipate claim 1.

Independent claims 10 and 12 contain limitations comparable to those of claim 1 and is allowable over Wallach et al. at least for these reasons.

Claim 10 further requires identifying the failing PCI slot from an address associated with a base address register when the logged error address falls within a known address size range for the address associated with that base address register and identifying the failing PCI slot as unknown when the logged error address fall after a known address size range of an address associated with that base address register preceding the logged error address. Claims 5 – 7, which depend indirectly from claim 1, contain comparable limitations as does independent claim 12.

The Examiner, in responding to applicant's arguments regarding why Wallach et al. failed to disclose these limitations, stated that "it is still the Examiner's position that the

address size must be within address space (32 bit, for example)." Applicant submits that the Examiner is misconstruing the limitations of these claims that require, for example with reference to claims 5 -7, "a known address size range of an address associated with that base address register." A known address size range of an address associated with a base address register is not the entire address range of the computer. Rather, as is known, it is a subset. As explained in paragraphs [0004] and [0005] of the present application:

> **[0004]** As is known, in a computer having multiple PCI slots, addresses are somewhat arbitrarily assigned to the PCI slots when the computer is powered-up. Also as is known, power-up software must build a consistent address map before booting the computer to an operating system. Thus, it must determine how much memory is in the system and how much address space the I/O controllers in the system require. This map, called a PCI resource allocation map, is a map of addresses that shows what addresses were assigned to the interface cards or I/O controllers in PCI slots during power-up. Once this is done, the power-up software can map the I/O controllers into reasonable locations and proceed with system boot.
>
> **[0005]** To do this mapping in a device independent manner, the base registers, called base address registers or BARs, are placed in a predefined header portion of the configuration register space in PCI compliant computers. BARs that map into memory space can be 32 bits or 64 bits wide while BARs that map into I/O space are always 32 bits wide. The number of upper bits that a device actually implements depends on how much address space to which the device responds. For example, a device that wants a 1 MB memory address space (using a 32 bit BAR) would build the top 12 bits of the address register, hardwiring the other bits to zero. By writing all 1's to a BAR associated with a device, the address space the device requires can be determined. The device will return 0's in all don't-care address bits (the lower address bits that have been hardwired to zero), and the size can thus be determined from the address bits returning zero. This is explained in more detail in Chapter 6 of the PCI Local Bus Specification, Revision 2.2, published by the PCI Special Interest Group, 2575 N. E. Kathryn # 17, Hillsboro, Oregon 97124.

An address of a failing PCI slot can thus fall within the address range of the computer, for example, 32 bits, yet not fall within any known address range associated with a BAR. And it is to this that these limitations of claims 5 – 7, 10 and 12 are directed – identifying a failing PCI slot from an address associated with a base address register when the logged error address falls within a known address size range associated with that base address register and identifying the failing PCI slot as unknown when the logged error address falls

after a known address size range of an address associated with that base address register preceding the logged error address. The logged error address in this case would not fall outside the address range of the computer, it would fall outside any known address size range associated with a BAR. Applicant thus submits that Wallach et al. fails to disclose these limitations and that claims 5 – 7, 10 and 12 are also allowable over Wallach et al. for these reasons also.

Claims 2 – 9 depend directly or indirectly from claim 1 and are allowable for at least that reason.

Claim 11 depends from claim 10 and is allowable for at least that reason.

Claim 13 depends from claim 12 and is allowable for at least that reason.

Finally, applicant submits that the Examiner is construing the claims more broadly than is consistent with the specification and the interpretation that those skilled in the art would reach. As the MPEP § 2111 states: "The broadest reasonable interpretation of the claims must also be consistent with the interpretation that those skilled in the art would reach." Applicant submits that those skilled in the art of computers would not construe the term "firmware" to mean any type of software, but only that software that is stored in ROM or PROM. Applicant also submits that those skilled in the art would not construe the limitation "known address size range of an address associate with that base address register" to mean the entire address range of the computer system, but only that address range allocated to the BAR.

## CONCLUSION

Applicant submits that all of the stated grounds of rejection have been properly traversed, accommodated, or rendered moot. Applicant therefore respectfully requests that the Examiner reconsider and withdraw all presently outstanding rejections. Applicant submits that a full and complete response has been made to the outstanding Office Action, and as such, the present application is in condition for allowance. Thus, prompt and favorable consideration of this amendment is respectfully requested. If the Examiner believes that personal communication will expedite prosecution of this application, the Examiner is invited to telephone the undersigned at (248) 641-1600.

Respectfully submitted,

Dated: JAN. 24, 2005

By: _____
Roland A. Fuller III
Reg. No. 31,160

HARNESS, DICKEY & PIERCE, P.L.C.
P.O. Box 828
Bloomfield Hills, Michigan 48303
(248) 641-1600

RAF/akb